Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Rok Zupančič

Vizualizacija zvokov na podlagi podobnosti

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V okviru naloge razvijte aplikacijo za vizualizacijo zbirke zvočnih posnetkov glede na njihovo medsebojno podobnost. Preučite stanje na področju vizualizacije zvokov, raziščite načine za izračun podobnosti med zvoki in uporabite gručenje za večnivojsko vizualizacijo, ki jo razvijte v obliki spletne aplikacije.

Izjava o avtorstvu diplomskega dela

Podpisani Rok Zupančič sem avtor diplomskega dela z naslovom:

Vizualizacija zvokov na podlagi podobnosti

S podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Matija Marolta,
- je elektronska oblika diplomskega dela, naslova (slov., ang.), povzetka (slov., ang.) ter ključnih besede (slov., ang.) identični s tiskano obliko diplomskega dela,
- soglašam z objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, 14. maja 2015 Podpis avtorja:

Študij brez pravih prijateljev, staršev in dobrih profesorjev gotovo ne bi potekal tako gladko, kot je. Zato se zahvaljujem vsem, ki ste mi stali ob strani, me spodbujali in podpirali. Posebna zahvala gre mentorju za strokovne napotke, pomoč in potrpežljivost, zaradi katerih je to diplomsko delo lahko ugledalo luč sveta.

Hvala!

Kazalo

Seznam uporabljenih kratic

Kratica	Angleško	Slovensko
CSS	Cascading Style Sheets	Kaskadne stilske podloge
DOM	Document Object Model	
D3	Data-Driven Documents	
HTML	Hyper Text Markup Language	
JSON	JavaScript Object Notation	
MFC	Mel-frequency cepstrum	Mel frekvenčni kepstrum
MIR	Music information retrieval	Pridobivanje podatkov iz glasbe
SVG	Scalable Vector Graphics	Umerljiva vektorska grafika
WWS	Work With Sounds	

Povzetek

Diplomska naloga govori o vizualizaciji zbirke zvočnih posnetkov po kriteriju medsebojne podobnosti. Produkt diplomskega dela je spletna aplikacija, ki vsebuje grafično vizualizacijo izračunanih podatkov in meta podatke opazovanih posnetkov. Delovno množico sestavljajo zvočni posnetki, ki smo jih pridobili s spletne strani projekta Work With Sounds ter nekaterih drugih. Najprej smo za vsak posnetek izračunali 27 nizkonivojskih značilk, na podlagi katerih smo določili podobnosti. Posnetke smo nato s hierarhičnem grozdenjem združili v gruče. Končno vizualizacijo smo izdelali z grafičnimi elementi SVG, ki jih na podlagi podatkov generira knjižnica D3 javascript. S slednjo smo realizirali tudi uporabniško interakcijo ter omogočili ogled slike zvoka in predvajanje posnetka. Vizualizacija temelji na predlogi "zoomable circle packing".

Ključne besede: zvok, vizualizacija, podobnost, d3.js.

Abstract

The thesis focuses on the visualization of a collection of sound recordings by the criterion of mutual similarity. The product of the thesis is an online application, which includes a graphic visualization of the calculated data and metadata of the observed recordings. The studied group consists of sound recordings, which we acquired from the website of the Work With Sounds project and a few other sites. First we calculated 27 low-level features of each recording, which were the basis on which we determined the similarities. Then we used hierarchical clustering to combine the recordings into groups. The final visualization was made with SVG graphic elements, which are based on data and generated by the D3 JavaScript library. We used the library to realise the user interaction and make it possible to see the picture of sound and play the recording. The visualization is based on the zoomable circle packing method.

Keywords: sound, visualization, similarity, d3.js.

Poglavje 1

Uvod

Ogromne količine neobdelanih podatkov same po sebi ne povedo veliko, zato je nujno nadaljnje procesiranje, h kateremu spada tudi vizualizacija. Težnja k razumljivosti in učinkoviti predstavitvi informacij vodi na široka znanstvena, statistična in druga informacijska področja. Kvalitetna predstavitev nam odpre nove dimenzije podatkov, iz katerih lahko tvorimo sicer težko opazne informacije. Na podoben način poskušamo na podlagi vizualizacije pridobljenih podatkov odkriti relacije oz. podobnosti med posameznimi zvočnimi posnetki. V raziskovalnem delu želimo poiskati podobnosti med industrijskimi zvoki, zato delovna množica vsebuje predvsem posnetke, ki so produkt sodelovanja nekaterih evropskih tehničnih muzejev [16]. Cilj diplomske naloge je izdelati spletno aplikacijo, ki prikazuje relacije med posameznimi zvočnimi posnetki. Na podlagi vizualizacije prepoznamo stroje, ki so si glede na izračunane podatke podobni oz. različni. S pridobivanjem podatkov s posnetkov pripravimo objektivno vizualizacijo, ki nam pomaga razbrati nove informacije in ugotovitve. Poleg tega je cilj naloge narediti zvočno zbirko bolj atraktivno in enostavno za preiskovanje. Hkrati smo v nalogi pozorni na to, da aplikacija dovolj dobro deluje tud na napravah z manjšimi zasloni.

Raziskovanje se prične s pridobivanjem podatkov iz omenjene množice. Pri tem si pomagamo s knjižnico ESSENTIA [5], ki nudi zbirko algoritmov za pridobivanje informacij iz zvokov. Računanje podobnosti med pridobljenimi

podatki ter generiranje primernega formata novih podatkov izvedemo z orodjem Matlab. Sledi grafična predstavitev novih podatkov v obliki spletne aplikacije. Vizualizacijo izvemo s pomočjo javascript knjižnice D3.js [4], ki poenostavlja delo z dokumenti, temelječimi na podatkih in skalabilno vektorsko grafiko SVG [15]. Cilj vizualizacije je predstaviti podobnosti med zvočnimi posnetki ter omogočiti na njej nekaj interakcije. Pridobljene podatke predstavimo z grafično predlogo, ki temelji na hierarhičnem drevesu. Da bi se izognili podatkovni zmešnjavi, podrobnosti na podlagi uporabnikovih akcij prikazujemo postopoma. Končna vizualizacija omogoča tudi predvajanje posameznih posnetkov, prikaz metapodatkov ter slike povzročitelja zvoka. Vse skupaj je ogrodjem Bootstrap [1] prilagodljivo napravam z različnimi velikostmi zaslonov ter dosegljivo na spletu.

V teoretičnem delu diplomske naloge najprej predstavimo zgodovino in načine podatkovne vizualizacije ter obstoječe vizualizacije zvočnih zbirk. Sledijo osnove zvoka ter opis pridobivanja podatkov iz zvokov. V prvem poglavju na kratko povzamemo tudi gručenje podatkov. Drugo poglavje je namenjeno predstavitvi razvojnih orodij, knjižnic in ogrodij, ki jih uporabimo v praktičnem delu. Sledi poglavje o praktičnem delu, kjer opišemo izbor in pripravo delovne množice, pridobljene podatke, potek računanja podobnosti, gručenje in pripravo izhodnega dokumenta. Poglavje zaključimo s programiranjem aplikacije in njenimi funkcionalnostmi.

Poglavje 2

Pregled področja

2.1 Vizualizacija

Vizualizacija je širok pojem, ki pomeni vizualno predstavitev oz. upodobitev nečesa. Ker je osrednja tema diplomske naloge vizualizacija podatkov pridobljenih iz zvokov, se v nadaljevanju osredotočimo na podatkovno vizualizacijo.

Podatkovna vizualizacija je grafična predstavitev podatkov v obliki, ki uporabniku omogoča kvalitativno razumevanje posredovane informacije [17]. Cilj vizualizacije je na enostaven in hkrati informativen način s pomočjo grafičnih elementov opisati oz. upodobiti podatke. S pomočjo vizualizacije lahko uporabnik hitro pridobi uporabne informacije, ki brez upodobitve niso opazne.

2.1.1 Zgodovina (podatkovne) vizualizacije

Prve vizualizacije je ustvaril že pračlovek z risanjem jamskih slik. Odtlej poskuša človek enostavno in razumljivo predstavljati podatke, kar je najlažje narediti z grafiko. Že 6.200 let pr. n. št. je nastal prvi znani zemljevid, v 6. stoletju pr. n. št. pa celo prvi zemljevid sveta. Sledilo je vse več poskusov vizualizacije nebesnih teles. Razvoj podatkovne vizualizacije se je močno pospešil v 17. stoletju, ko so natisnili prve astronomske slike, nastale so tabele

logaritmov, koordinatni sistem, razvijala se je statistika itd. V 18. stoletju zemljevidi dobili nove razsežnosti. Geografi so jih obogatili z dodatnimi vrednostmi, na primer barvami in črtami. Vizualizacija se je razvila tudi v ekonomiji in geografiji. Grafikoni in histogrami, kot jih poznamo danes, so se razvili v prvi polovici 19. stoletja. Zelo so napredovale statistične analize, ki so vse bolj težile k vizualni predstavitvi podatkov. Medtem ko je prva polovica 20. stoletja temna doba vizualizacije, saj ni bilo posebnih napredkov, se je proti koncu druge polovice in z nastankom prvih računalnikov razvoj ponovno pospešil.

2.1.2 Načini vizualizacije

Obstaja mnogo načinov podatkovne vizualizacije. Ben Shneiderman deli vizualizacijo v 7 kategorij [22]. **Enodimenzionalna** vizualizacija je linearni seznam podatkov, urejen po določenem kriteriju. Ta način vključuje tudi oblikovanje besedila in metode preiskovanja po seznamu. **Dvodimenzionalna** vizualizacje združuje ravninske predstavitve oz. zemljevide, na katerih so poudarjena določena področja v odvisnosti med dvema domenama. **Tridimenzionalni** način vizualizacije predstavljajo objekti iz realnega sveta. Tu imajo pomembno vlogo prostornina, površina in položaj objektov v prostoru. **Večdimenzionalno** vizualizacijo dosežemo z manipulacijo podatkov. To lahko storimo z dodatnimi atributi, ki jih naprimer spreminjamo z drsniki ter v realnem času vplivamo na izgled vizualizacije. Poseben način vizualizacije so **drevesa**, ki prikazujejo podatke v hierarhičnem razmerju (binanrno, krožno, hiperbolično). **Kronološke** vizualizacije obravnavajo podatke v odvisnosti od časa. Relacije med podatki odlično prikazujejo **mrežni** načini (matrike, aluvialni diagram, relacijski diagram).

2.1.3 Vizualizacija zvočnih zbirk

Z vizualizacijo zvočnih zbirk se je z različnimi cilji ukvarjalo že veliko raziskovalcev. Želeli so omogočiti enostavno brskanje po zvočnih zbirkah, umestiti

posnetke v geografski prostor, prikazati relacije med posnetki, posamezniku priporočiti skladbo itd. Dosedanje delo na področju vizualizacije zvočnih zbirk je z obširno raziskavo predstavila Anita Shen Lillie [20]. Osnovna vizualizacija zvočne zbirke je tekstovni seznam posnetkov, ki ga običajno vidimo v predvajalnikih glasbe, naprednejše pa že vsebujejo grafične elemente; smiselne povezave med njimi določijo pomen vizualizacije. Ker so bile vizualizacije razvite s točno določenimi cilji, je pomen povezav v posameznih primerih različen. Primera enostavnih grafičnih vizualizacij sta LivePlasma [8] in Musicovery [11]. Z razširitvijo podatkovnih dimenzij postanejo osnovne grafične vizualizacije bolj nazorne. Primer so zemljevidi podatkov nepTune [12] in PlaySOM [13], ki na ravnini oz. v prostoru z barvnimi vzorci upodobijo zvočno zbirko. Preglednost vizualizacij se s povečevanjem podatkovnih dimenzij zmanjša, zato se uveljavijo interaktivne vizualizacije. Interaktivnost omogoča oglede specifičnih delov zvočnih zbirk ter prikrivanje podrobnosti. Primer takšne vizualizacije sta MusicBox [20] in Formater [6].

Poseben primer vizualizacije zvočnih zbirk so priporočilni sistemi. Uporabnik vnese v sistem začetne informacije (skladbo, izvajalca, žanr), sistem pa mu vrne priporočene, podobne ali kako drugače povezane vsebine. Primer takšne vizualizacije je MusicMap [10].

Naša naloga govori o vizualizaciji podobnosti zvokov, zato se v nadaljevanju osredotočimo na vizualizacije, ki so potencialno najboljše za prikazovanje relacij med podatki. Iščemo torej relacijske vizualizacije, med katerimi so v ožjem izboru:

- točke v večdimenzionalnem prostoru (človeku primerne so 1D–4D),
- hierarhično drevo, ki je lahko predstavljeno na klasičen način ali v obliki vgnezdenih krogov,
- hiperbolični krog, kjer so na krožnici razporejeni zvoki, v središču kroga pa črte različnih barv in velikosti označujejo podobnost,
- podobnostne povezave med dvema seznamoma posnetkov.

2.2 Zvok

Zvok je mehansko valovanje, ki se širi v dani snovi. Valovanje je vzdolžno razen v trdnih snoveh, kjer se lahko sproži tudi prečno valovanje. Za širjenje potrebuje medij, kar pomeni, da se v vakuumu ne širi. Opišemo ga s frekvenco (višina) in amplitudo zvočnega tlaka (glasnost). Človeški slišni spekter je med 20Hz in 20kHz. Zvoki z nižjo frekvenco so infrazvoki, zvoke z višjo frekvenco pa imenujemo ultrazvoki.

Človeški organ oz. čutilo za zaznavanje zvoka je uho, ki ima tri funkcionalne enote, in sicer zunanje, srednje in notranje uho. Posamezne enote so sestavljene iz podenot. Zunanje uho tako sestavljata uhelj, ki lovi zvočno valovanje, ter sluhovod, ki deluje kot resonator, preko katerega se valovanja prenesejo v srednje uho. Srednje uho sestavljajo bobnič, koščice kladivce, nakovalce in stremence ter evstahijeva cev. Tudi srednje uho ojačuje in prenaša zvočne vale v notranje uho. V notranjem ušesu se nahajata polž in slušni živec. V polžu, ki je poln slušnih dlačic in tekočine, se zvočni valovi pretvorijo v živčne impulze in se po slušnem živcu prenesejo v možganski slušni center. V ušesu se nahaja tudi čutilo za ravnotežje.

2.2.1 Pridobivanje podatkov z zvočnih posnetkov

Pridobivanje informacij iz zvoka se je pričelo v 50. letih prejšnjega stoletja, do pomembnih premikov pa je prišlo v 70ih letih. Cilji raziskav so bili različni - avtomatično prepoznati glasbo in jo zapisati v simbolno obliko, prepoznavati instrument, zgolj vizualizirati posamezen zvok, poiskati podobnosti med več zvoki itd. Takrat je bilo ustanovljeno tudi več raziskovalnih institucij [19].

MIR oz. Music Information Retrieval je Stephen Downie zasnoval [26] kot multidisciplinarno raziskavo, da bi razvil način iskanja po vsebini posnetkov ter nato ustvaril največjo in prosto dostopno glasbeno zbirko. Področje se je močno razširilo in vključuje vse od muzikologije in psihologije do informacijskih znanosti, bibliotekarstva in mnogo drugih.

Knjižnice in orodja [25], ki omogočajo izračun značilk, so v sklopu orodja

Matlab Mirtoolbox, spletna storitev EchoNest, MusicMiner, RapidMiner, C++ knjižnica CLAM in ESSENTIA itd. Slednjo smo uporabili tudi v tem diplomskem delu in jo poleg pridobljenih podatkov predstavljamo v nadaljevanju.

2.3 Gručenje podatkov

Gručenje je način razvrščanja podatkov za potrebe samostojnega reševanja problemov. Uporablja se pri strojnem učenju, podatkovnem rudarjenju, napovedovanju, statistiki itd. Pri tem velja načelo, da sta si objekta v isti skupini bolj podobna, objekta iz različnih skupin pa različna. Gručenje ni zgolj specifičen algoritem, temveč širši problem, ki ga razrešujemo z različnimi metodami.

Osnovna delitev gručenja je hierarhična in delitvena tehnika [18]. Med hierarhične tehnike spadajo metoda enojnega povezovanja, celostno povezovalna metoda, metoda povezovanja glede na povprečje, metoda mediane itd. Delitvene tehnike so k-means, c-means, metoda povezovanja glede na povprečje itd.

Hierarhične metode temeljijo na razdruževanju ali zaporednem združevanju dveh ali več gruč v novo. Najvišji nivo predstavlja gruča z vsemi elementi, na najnižjem nivoju pa je gruča kar vsak posamezni element. Rezultat gručenja nazorno prikaže drevo združevanja oz. dendrogram, ki običajno nastane pri zaporednem združevanju najbolj podobnih elementov. Druga skupina metod gručenja so delitvene oz. nehierarhične. Pri tem uporabnik določi kriterij delitve ter poskuša minimizirati vrednost kriterijske funkcije. Delitvene metode predstavljajo lokalno optimalno delitev. Za optimalno rešitev jih je priporočljivo uporabiti večkrat. Pri vseh metodah gručenja velja pravilo, da posamezen element hkrati pripada samo eni gruči.

Gručenje podatkov, prodobljenih iz zvoka, poteka popolnoma enako. Na končni rezultat vplivajo izbira metode gručenja, način pridobivanja podatkov s posnetkov ter nabora značilk.

Poglavje 3

Razvojna orodja in knjižnice

3.1 Orodja

3.1.1 Matlab

MATLAB [9] je programsko orodje za numerično analizo ter tudi programski jezik četrte generacije, kar pomeni, da je primeren za hitro izdelavo uporabniških programov. Omogoča enostavno delo z matrikami in vektorji. Primeren je tudi za risanje funkcij, implementacijo algoritmov, analizo slik, obdelavo digitalnih signalov, analizo in razvoj vodenja sistemov, načrtovanje filtrov itd. S pomočjo dodatnih knjižnic oz. toolbox- ov je odličen pripomoček pri zgoraj naštetih opravilih. Matlab omogoča tudi uporabo dodatnih knjižnic, napisanih v drugih programskih jezikih, kot so C, C++, JAVA, Fortran, Python. Prav tako je Matlab možno klicati iz drugih programskih jezikov.

Razvoj Matlaba se je pričel na začetku 80. let prejšnjega stoletja. Cilj je bil študentom omogočiti uporabo že razvitih knjižnic za izvajanje linearne algebre ter numerično računanje brez potrebnega predhodnega programerskega znanja. Združili so moči Cleve Moler, Jack Little in Steve Bangert in ustanovili podjetje MathWorks. Njihov izdelek so najprej uporabljali v nadzornem inženirstvu, kmalu pa se je razširil tudi v izobraževalne in razi-

skovalne ustanove.

3.1.2 Brackets

Brackets [2] je razvojno orodje, ki smo ga uporabili za kodiranje. Je odprtokodni urejevalnik za spletni razvoj, s poudarkom na dizajnu. Nudi množico vizualnih orodij, ki razvijalcu oz. spletnemu oblikovalcu poenostavijo delo. Med drugim omogoča „inline" urejanje kode, predogled v brskalniku v realnem času, podpira preprocesorje LESS in SCSS, omogoča uvoz datotek Photoshop za enostavnejše spletno oblikovanje in vrsto drugih razširitev. Orodje je nastalo pod okriljem podjetja Adobe z licenco MIT.

3.2 Knjižnice

3.2.1 ESSENTIA

ESSENTIA [5] je odprtokodna knjižnica za analizo zvoka, napisana v jeziku C++. Ščiti jo licenca Affero GPLv3. Nudi zbirko algoritmov za računanje podatkov oz. značilk iz zvokov. Knjižnico je mogoče uporabiti znotraj Python skript na podoben način, kot lahko nekatere knjižnice, napisane v drugih jezikih, uporabljamo v Matlabu. Essentia omogoča pretočni način (ang. streaming mode), kjer lahko algoritme povežemo in avtomatično izvajamo. Tako prihranimo čas izvajanja, porabimo manj pomnilnika in se izognemo ponavljajočim se delom kod. S pomočjo knjižnice lahko:

- beremo in zapisujemo avdio datoteke,
- procesiramo signale (FFT, DCT, frame cutter, windowing, envelope, smoothing),
- filtriramo zvoke po frekvenci, glasnosti itd.,
- računamo statistične podatke (mediano, povprečje, varianco, sploščenost),
- računamo časovne opisnike (trajanje, glasnost, zero-crossing-rate),

- računamo spektralne opisnike (spektralne vrhove, inharmoničnost, disonanco),
- računamo tonaliteto (višino tona, glavno melodijo, kontrast, uglasitev),
- računamo ritmične opisnike (prepoznavanje ritma, hitrost ritma, note), in
- računamo druge visoko - nivojske opisnike (dinamičnost, segmetacijo, dancability).

Essentia je torej zbirka algoritmov z dodano sposobnostjo izvajanja več procesov hkrati in nižjo porabo spomina. Lastnosti so usmerjene v robustnost, zmogljivost in optimalno izvajanje vključenih algoritmov, hkrati pa je enostavno razširljiva z novimi algoritmi. Knjižnica ima tudi podporo za aplikacijo Sonic visualiser [14], s pomočjo katere je možno vizualizirati vsebino zvoka. Sicer knjižnica podpira več platform - operacijske sisteme Linux, Windows in Mac OS.

3.2.2 Knjižnica D3.js

D3.js je javascript knjižnica za manipulacijo dokumentov, ki temeljijo na podatkih. Namenjena je vizualizaciji podatkov. Kot podporne tehnologije uporablja spletni označevalni jezik HTML, vektorske grafične elemente SVG in oblikovalni jezik CSS. Z omenjenimi standardi knjižnica dosega kompatibilnost z sodobnimi brskalniki brez dodatnih omejitev pri vizualizaciji. S knjižnico D3 lahko podatke povežemo na DOM,[1] nato pa nad njimi izvajamo želene transformacije.

Zametki knjižnice segajo v leto 1996, ko je podjetje Netscape izdelalo prvi brskalnik z možnostjo uporabe javascripta. Sledil je razvoj raznih orodij za

[1] Document Object Model je standard, neodvisen od jezika in platforme, ki določa, kako so objekti predstavljeni na spletni strani. Definira povezave med atributi in objekti ter opisuje, kako se spreminjajo atributi.

vizualizacijo s pomočjo flash in javascript knjižnic, D3 pa nastane leta 2009 [24].

Vizualizacija podatkov se s pomočjo knjižnice D3 izvede v treh korakih. Prvi je nalaganje podatkov v brskalnik, sledi povezovanje podatkov z grafičnimi elementi, na koncu pa transformacija oz. nastavitev atributov grafičnih elementov. Podatke, ki jih uporablja knjižnica, lahko pridobimo iz vgrajenih tabel v kodi, naložimo iz podprtih dokumentov (JSON, CSV, TSV, XML, TXT, HTML) ali jih pridobimo preko zahtevka http (XMLHttoRequest).

Vizualizacijo izdelamo z grafičnimi elementi SVG, njihov izgled pa z jezikom CSS. Pomemben del risanja elementov s knjižnico D3 je pripenjanje pravilnih atributov. Atribute statične grafike nastavimo v naprej, atribute dinamični pa preračunamo s transformacijo. Šele s pravilnimi atributi dosežemo, da se bo element prikazal v vsebovalniku SVG.

3.2.3 SVG

SVG (scalable vector graphic) je odprtokodni standard, ki ga uporabljamo za izdelavo dvodimenzionalnih vektorskih grafik. Omogoča nam izdelavo zelo kompleksnih grafik in profesionalnih načrtov, obenem pa je njegovo strukturo v osnovi zelo lahko za razumeti, saj temelji na označevalnem jeziku XML [23]. Za uporabo ga je W3C (Mednarodni inštitut za razvoj spletnih standardov) predlagal septembra 2001.

Grafiko SVG definira značka <svg>, ki ustvari polje, v katerega nizamo posamezne elemente ter jim preko atributov določimo lastnosti.

Prednosti SVG [15]:

- grafiko lahko ustvarimo in urejamo v tekstovnem urejevalniku,
- po elementih SVG lahko iščemo, jih indeksiramo in kompresiramo,
- grafika je skalabilna,
- grafika se lahko izriše v visoki kvaliteti v vsaki resoluciji,

- grafiko lahko poljubno povečujemo ali zmanjšujemo,
- je odprtokoden standard,
- datoteke so čisti XML.

SVG podpira tudi animacijo. Z nekatgerimi elementi lahko spreminjamo vrednosti posameznih atributov v času, nastavljamo nove vrednosti, premikamo elemente, animiramo barvo ali spreminjamo pozicijo in rotacijo.

3.2.4 HTML5

HTML (hyper text markup language) je osnovni in hkrati standardni označevalni jezik spletnih strani in aplikacij. Z njim določimo prikaz spletnih dokumentov v brskalniku, strukturo in semantični pomen posameznih delov. Jezik HTML so torej značke, ki definirajo dokument, glavo, telo, naslove, odstavke in nove vrstice, sezname, sidra, slike, tabele itd. Običajno nastopajo v paru - začetna in končna.

Posebno označevanje dokumentov se je začelo leta 1980 zaradi potrebe po oblikovanju in prenašanju dokumentov med raziskovalci. Deset let pozneje je idejni znanstvenik Tim Berners-Lee definiral HTML ter razvil brskalnik in strežniško opremo, ki omogoča branje tako označenega dokumenta. Kot svetovni standard je bil uradno priznan leta 2000.

Pomemben preskok je pomenila nova verzija imenovana HTML5, ki so jo popolno in končno predstavili oktobra 2014. Nekatere izmed novosti so [21]:

- standard za prikaz video in avdio vsebine, ki ga v prejšnjih verzijah še ni bilo;
- strukturne značke, ki nadomeščajo nekatere oznake in bolj specifično opisujejo vsebino;
- razširjeni spletni obrazci z novimi vhodnimi tipi in dodatnimi atributi;
- funkcija primi in spusti (drag&drop), ki omogoča uporabniku prijemanje virtualnega objekta in prenos na drugo lokacijo;

- grafični objekt *canvas*, ki omogoča dinamično risanje oz. prikazovanje grafike na spletnih straneh s pomočjo javascripta.

3.2.5 CSS

CSS (cascading style sheets) je slogovni jezik, ki se uporablja za opisovanje videza in postavitve elementov v DOM [3]. S to sintakso elementom določimo barvo, velikost, odmik, poravnavo, obrobo, pozicijo, pisavo itd. Sintaksa je preprosta in sestoji iz angleških besed. Slog lahko pišemo v ločenih datotekah, ki jih vključimo v dokument, ga vgradimo med značke <style></style> ali vsakemu posameznemu elementu pripišemo atribut *style*, ki vsebuje CSS. Pri večjih projektih je priporočljiv predvsem prvi način, saj tako ohranimo preglednost dokumentov, ne ponavljamo kode in tako zmanjšamo velikost datotek, poenostavimo odkrivanje napak itd.

Potreba po podrobnem oblikovanju dokumentov se je pojavila s širitvijo interneta v začetku devetdesetih let. Prvo uradno različico standarda sta leta 1996 izdelala Hakon Wium in Bert Bos in jo že dve leti pozneje nadgradila v drugo. Različico 2.1 so priznali šele leta 2011, razvoj tretje pa še poteka.

CSS, vključno z verzijo 2.1, podpira večina brskalnikov, medtem ko verzija 3 še ni povsem podprta. Sistemi, ki skrbijo za interpretacijo in povezovanje elementov s slogovnimi datotekami, so različni, zato včasih prihaja do težav s konsistentnostjo. Težava predstavljajo preglavice razvijalcem, ki morajo poskrbeti, da se bo dokument pravilno prikazoval neodvisno od interpretatorja.

Poglavje 4

Razvoj aplikacije

4.1 Priprava delovne množice

Preden začnemo obdelavati podatke, moramo pripraviti delovno množico. Del zvočnih posnetkov pridobimo s spletne strani Work With Sounds[1] [16], preostali del pa iz mentorjeve delovne množice. Na omenjeni spletni strani je mogoče najti tudi slike povzročitelja zvoka, druge pa poiščemo na spletu. Pripravimo si množico 295 zvokov in 295 slik. Da bi bilo nalaganje spletene strani optimalno, ustvarimo novo množico slik, ki jih z urejevalnikom fotografij Photoshop pomanjšamo. Manjše slike se bodo naložile hkrati s spletno stranjo, večje pa lahko uporabnik prikliče s klikom na manjšo.

4.2 Pridobivanje podatkov iz zvočnih posnetkov

Prvi korak izdelave diplomske naloge je pridobivanje podatkov iz zbirke zvokov. Zbirka obsega 295 posnetkov. S spletne strani Work With Sounds

[1] Evropski projekt v katerem sodeluje šest tehniških muzejev na Poljskem, v Sloveniji, Nemčiji, na Finskem, v Belgiji in na Švedskem. Ti muzeji ohranjajo industrijsko dediščino Evrope, namen njihovega sodelovanja pa je posneti in dokumentirati dediščino izginjajoče industrijske družbe.

pridobimo 95 posnetkov, v mentorjevi zbirki pa 200. Za pridobivanje podatkov uporabimo odprtokodno knjižnico Essentia, s katero izračunamo 27 nizkonivojskih značilk za vsak posnetek.

4.2.1 Značilke

- Razpon na lestvici Bark (ang. bark bands spread) je značilka, ki pove spektralno energijo posameznih pasov lestvice Bark.[2]

- Spektralni centroid (ang. spectral centroid) je vrednost, ki opisuje geometrično središče frekvenčnega spektra. Izračunamo ga kot uteženo povprečno vrednost frekvenc, ki jih pridobimo s Fourierovo transformacijo, uteži pa predstavlja magnituda frekvenc. Spektralni centroid predstavlja svetlost zvoka.

- Spektralna energija (ang. spectral energy band) je značilka, ki prikaže energijo spektra v danem frekvenčnem pasu. V nalogi izračunamo energijo za nizke, nizke-srednje, srednje, srednje-visoke in visoke frekvence.

- Mel kepstralni koeficienti – MFCC (ang. mel-frequency cepstrum) opisujejo spekter signala. Temeljijo na linearni kosinusni transformaciji logaritmov spektra moči in na nelinearni lestvici melfrekvenc.

- Spektralni kontrast (ang. spectral contrast) je značilka, ki opisuje spektralne lastnosti signala. Opisuje razliko v moči signala med vrhovi in dolinami. Pri računanju na posameznem posnetku z različnimi parametri algoritem izvedemo večkrat.

- Stopnja menjavanja akordov (ang. chords changes rate) je opisnik, ki nam pove stopnjo oz. hitrost menjavanja akordov. Značilka je del opisnikov tonskih lastnosti.

[2] lestvica Bark je psihoakustična lestvica, ki opisuje subjektivno zaznavanje glasnosti. Obsega prvih 24 kritičnih pasov sluha od 0 do 15,5 kHz.

4.3 Računanje podobnosti

V procesu pridobivanja podatkov iz zbirke zvokov za vsak posnetek izračunamo vektor, ki sestoji iz zgoraj opisanih značilk (nekatere so z različnimi parametri izračunane večkrat). V naslednjem koraku z orodjem Matlab nad vektorji izračunamo podobnosti. Tako pridobimo simetrično matriko dimenzij 295 x 295.

Za računanje podobnosti uporabimo funkcijo *pdist*, ki nam izračuna kosinusno razdaljo med dvema vektorjema. Funkcija sicer omogoča računanje razdalj tudi po drugih metodah, kot so evklidska, Hemmingova, korelacijska itd. Iz matrike, ki so jo sestavljali vektorji značilnic tako pridobimo vektor razdalj. V tem delu naloge izračunani vektor pretvorimo v kvadratno simetrično matriko, saj je matrika tako primernejša za nadaljnjo obdelavo.

4.3.1 Kosinusna razdalja

Kosinusna razdalja je kot med dvema vektorjema na ravnini, ki jo opisujeta. Razpon podobnosti je med 0 in 1. Z 0 predstavimo identična vektorja, z 1 pa popolnoma različna. Iz tega sklepamo, da manjši ko je kot, bolj podobna sta si vektorja. Prednost kosinusne razdalje je, da ni odvisna od dolžine vektorjev.

4.4 Gručenje in izdelava dokumenta JSON

Grozdenje oz. gručenje [7] in izdelava JSON dokumenta je daljši in zelo pomemben postopek. Izvedemo ga po metodi hierarhičnega grozdenja (ang. hierarchical clustering). Metoda združuje podatke v hierarhično drevo preko različnih lestvic. Drevo je večstopenjska hierarhija, kjer so grozdi na nižji stopnji hkrati del grozdov na višji. To nam omogoča, da poljubno določimo stopnjo grozdenja.

Postopek hierarhičnega grozdenja ima trei dele:

- iskanje podobnosti med posameznimi objekti, v našem primeru med

zvoki. Ta del algoritma smo že izvedli v prejšnjem koraku, kjer je bil rezultat simetrična matrika;

- grupiranje objektov oz. zvokov v hierarhično drevo;
- določitev rezanja hierarhičnega drevesa. V tem koraku določimo stopnjo, na podlagi katere bodo kreirane skupine objektov. Po rezanju dobimo več manjših dreves, ki sestavljajo posamezen grozd.

4.4.1 Gručenje

Povezovanje objektov v drevo

Simetrično matriko, ki jo pridobimo v prejšnji fazi izdelave aplikacije, sestavljajo objekti, katere v tem delu povežemo med seboj. Korak za korakom združujemo najbolj podobna objekta. Nastajajo novi pari, vse dokler ne pridobimo hierarhičnega drevesa. Gradnjo drevesa v celoti izvedemo z Matlab funkcijo *linkage*. Rezultat povezovanja objektov v hierarhično drevo je matrika dimenzij n x 3. Prvi in drugi stolpec predstavljata objekta, ki sta bila povezana, tretji pa razdaljo med njima. Vrstice matrike so urejene od najkrajše do najdaljše razdalje.

Pridobljeno hierarhično drevo po končanem postopku izrišemo, saj tako najlažje ocenimo rezultat dela. Izris izvedemo z Matlab funkcijo *dendrogram*, pri čemer ugotovimo, da je drevo zelo neuravnoteženo.

Uravnoteženje drevesa

Povezovanje objektov v drevo v osnovi povezuje najbližje objekte. Pri tem nastane zelo neuravnoteženo drevo, ki ni najbolj primerno za rezanje in določanje grozdov. Tako pri povezovanju objektov upoštevamo tudi razdaljo med grozdi, in sicer vsoto kvadratov razdalje od elementov do centroida grozda. Gradnjo uravnoteženega drevesa izvedemo po metodi Ward, ki jo kot atribut podamo funkciji *linkage*. Končni rezultat povezovanja je hierarhično drevo z enakomerno porazdeljenimi elementi.

Gradnja grozdov

Gradnja grozdov je zadnja faza gručenja, v kateri iz predhodno procesiranih podatkov dobimo dejanske grozde. Glede na velikost delovne množice in zaradi preglednosti se odločimo, da bodo posamezni nivoji vsebovali po štiri grozde, razen zadnji, ki bo vseboval tri.

Rezanje drevesa se prične pri korenu in poteka proti listom. Vsak rez nam vrne toliko grozdov, kolikor smo določili na začetku faze (meja rezanja se določi samodejno). Postopek ponavljamo na vseh poddrevesih, dokler nam ne zmanjka vozlišč. V našem primeru rezanje ponovimo štirikrat in tako dobimo štiri nivoje grozdov. Rezultat grozdenja je vektor, ki ga sestavljajo indeksi grozdov. Pozicija indeksa grozda v vektorju je identifikacijska številka objekta.

Opisani postopek nam izvede funkcija *cluster*, ki sicer podpira tudi drugi način rezanja. V drugem načinu ne določimo števila grozdov, temveč mejo rezanja, ki jo najlažje razberemo na dendrogramu. Funkcija nato na podlagi meje vrne različno število grozdov.

4.4.2 Izdelava dokumenta JSON

Izdelava dokumenta JSON prav tako poteka v okolju Matlab. Čepav obstaja knjižnica JSONlab, ki omogoča branje in pisanje dokumentov JSON, tokrat dokument kreiramo zgolj z osnovnimi funkcijami za delo z datotekami.

Generiranje dokumenta poteka hkrati z gručenjem. Hierarhično drevo grozdov iterativno, glede na število nivojev in število grozdov posameznega nivoja, preiskujemo po načelu preiskovanja v globino. Postopno obravnavamo posamezne grozde in jih delimo do globine 4. Izpisovanje objektov izvedemo na najglobljem nivoju, strukturni podatki dokumenta pa se izpisujejo tudi na višjih. Objekt sestavljajo identifikacijska številka, naslov, spletna povezava, obdobje, trajanje, število kanalov, frekvenca vzorčenja, bitna globina in glasnost.

4.5 Programiranje spletne aplikacije

Doslej opravljeno delo dejansko pride do izraza v naslednjem koraku. Kot nakazuje že naslov poglavja, je vizualizacija nekakšna aplikacija, dosegljiva na spletu. Aplikacijo razvijemo s knjižnico D3.js in elementi SVG. Pri razvoju si pomagamo tudi z jezikom HTML5 in CSS ter ogrodjem Bootstrap. Vizualizacijo razvijemo z orodjem Brackets.

Programiranje spletne aplikacije lahko razdelimo na dva dela. Prvi je običajno ogrodje HTML5, ki je temelj aplikacije, drugi pa skripta, ki je ključni del vizualizacije in je vgrajena v prvi.

4.5.1 Ogrodje HTML5

Ogrodje HTML5 nekakšna lupina, ki omogoča postavitev aplikacije na spletnem strežniku. Ogrodje sestavljajo običajni strukturni elementi HTML in je v grobem razdeljeno na glavo in telo.

Glava vsebuje metapodatke, in sicer način kodiranja znakov, ključne besede, kratek opis vizualizacije, naslov in ime avtorja. V tem delu vključimo tudi vse potrebene elemente ogrodja Bootstrap, ki nam omogoča pravilno prikazovanje strani na napravah z različnimi zasloni. Sem spadajo tudi vse slogovne datoteke, ki opisujejo obliko in postavitev uporabljenih elementov.

Ogrodje HTML se nadaljuje s telesom. Telo je tisti del dokumenta, ki ga brskalnik prikazuje na zaslonu. Smiselno ga razdelimo na posamezne sekcije, končno postavitev in obliko sekcij pa preko atributa *class* definiramo z ogrodjem Bootstrap. Telo se začne z navigacijskim menijem; ta vsebuje naslov aplikacije, povratno povezavo in gumb, ki odpre okno z nekaj informacijami.

Sledi vsebovalnik vizualizacije, ki je razdeljen na levi in desni del. Levi, večji del vsebuje vizualizacijo SVG, desni pa tabelo s podatki o izbranem zvoku, sliko in predvajalnik zvoka ter povezavo na zunanjo stran, če je na voljo. Slika predstavlja povzročitelja izbranega zvoka (predmet ali živo bitje) in se glede na uporabnikovo akcijo nad vizualizacijo ustrezno spreminja. Ker

je v osnovi v majhnih dimenzijah, smo aplikaciji dodali možnost povečave. Tako klik nanjo odpre pojavno okno, ki vsebuje sliko originalne velikosti.

Pod sliko se nahaja predvajalnik zvoka, ki omogoča, da uporabnik po kliku na objekt sliši vsebino. Predvajalnik je nov element HTML5 in je definiran na naslednji način:

```
1 <audio id="player" controls="controls">
2   <source id="mp3_src" src="zvok001.mp3" type="audio/mp3">
3 </audio>
```

Predvajalniku sledi dinamična tabela podatkov. V osnovi prikazuje ime (Name), trajanje (Duration), število kanalov (Channels), frekvenco vzorčenja (Sample rate), bitno hitrost (Bit rate), bitno globino (Bit depth), glasnost (Level) in interno identifikacijsko številko zvoka (ID). Ker smo množico vzorcev združili iz dveh, nimamo vedno na voljo vseh navedenih podatkov. Tako se pri delu posnetkov v informacijski tabeli prikazujeta samo ime in identifikacijska številka.

Tabeli sledi gumb s povezavo na zunanje spletno mesto, ki je omogočen le takrat, ko je povezava dejansko na voljo. Povezava je na voljo pri množici zvokov iz projekta WWS [16] in vodi na originalno stran posnetka.

Telo aplikacije se nadaljuje s kodo za pojavno okno, ki vsebuje nekaj osnovnih informacij o vsebini in pomenu diplomske naloge ter kratka navodila za uporabo. Pojavno okno odpremo s klikom na informacijsko ikono, ki se nahaja na desni strani glavnega menija.

Dobra praksa programiranja spletnih aplikacij je, da se čim več datotek javascript vključi na konec dokumenta. To omogoča, da se stran hitreje izriše na ekranu, saj v prvi sekundi vstopa na spletno stran funkcije niso pomembne in smo osredotočeni le na grafično podobo. Tako imajo skripte nekaj več časa, da se pravilno naložijo. Tik pred zaključno značko telesa vključimo knjižnice jQuery,[3] Bootstrap, Audioplayer[4] in Fancybox.[5] Poleg naštetih vključimo tudi tudi svojo zbirko funkcij javascript.

[3] Ena izmed knjižnic javascript, ki jo za delovanje potrebuje ogrodje Bootstrap.
[4] Skripta, ki jo za pravilno delovanje potrebuje predvajalnik zvokov.
[5] Skripta za odpiranje pojavnega okna s fotografijo.

4.5.2 Skriptiranje d3.js

Ključni del diplomskega dela je vizualizacija, ki je vključena v ogrodje HTML5. Vizualizacijo gradimo na predlogi „zoomalebe circle packing“ in z njo povezano knjižnico D3.js. Vizualizacijo lahko dodatno razdelimo na šest delov, ki jih s primeri predstaviljamo v nadaljevanju.

Definicija spremenljivk

Vizualizacija za delovanje uporablja precej globalnih spremenljivk in objektov. Tako na začetku definiramo velikost, postavitev, osnovno barvo in nekaj drugih spremenljivk. Definiramo tudi najpomembnejši spremenljivki, ki določita vizualizacijo. To sta vsebovalnik *svg* in razporeditev elementov *pack*. Nekaj njunih lastnosti določimo že s prvimi atributi.

Branje podatkov

Sledi branje podatkov, ki smo jih pripravili v obliki dokumenta JSON. Branje izvedemo z definicijo funkcije *readFile(fileName)*. Prebrane podatke nato shranimo v prej definirano spremenljivko, počistimo vsebovalnik ter kličemo funkcijo, ki začne z risati elemente. V tem koraku dodamo tudi pomožni gumb za prehod nazaj, katerega pomen in delovanje predstavimo v nadaljevanju. Morebitne napake pri branju sporočimo v konzolo brskalnika.

Risanje elementov

S postopkom risanja oz. dodajanja elementov v vsebovalnik podatke iz tekstovne oblike spremenimo v grafiko. Najprej izračunamo drevesno strukturo (tree layout). Knjižnica D3 za ta namen nudi funkcijo *nodes(source)*, ki za vsako vozlišče vrne strukturo z atributi *parent, children, value, depth, x, y,* in *r*.

Sledi ustvarjanje krogov oz. vozlišč vizualizacije. Posamezno vozlišče sestavljajo elementi <g>, ki jih pripenjamo na risalno površino. Vsakemu

vozlišču preko atributov določimo razred in ime ter vanje vgnezdimo kroge, ki v končni fazi predstavlja grafično podobo.

```
<svg width="729" height="729" id="svgId">
  <g transform="translate(364.5,364.5)">
    <g class="node node--root" name="-" style="">
      <circle transform="translate(0,0)" r="359.5" style="
          fill: rgb(118, 225, 209);"></circle>
    </g>
    <g class="node parent" name="skupina100">
      <circle transform="translate(-18.26,110.90)" r="
          101.71" style="fill: rgb(76, 204, 212);"></circle
          >
    </g>
    <g class="node parent" name="skupina110">
      <circle transform="translate(1.00,139.70)" r="17.17"
           style="fill: rgb(46, 181, 210);"></circle>
    </g>
    ...
    ...
    ...
  </g>
</svg>
```

Tako na podlagi podatkov ustvarimo vsa vozlišča, ki v brskalniku sicer še niso vidna. Vidnost dosežemo s transformacijo, ki elementom določi dodatna atributa položaja in velikosti.

Sledi pripenjanje slik na vozlišča. Najprej izberemo vse liste drevesa (elementi <g>), nato pa jim dodamo ustrezno sliko. Izbiranje vozlišč poteka na podlagi razredov, ki smo jih določili v predhodnem koraku. To storimo z vgrajeno funkcijo *selectAll*. Pripetim slikam dodamo še nekaj atributov, med katerimi je tudi *clip-path*, ki sliko poveže s samostojnim elementom *clipPath* ter tako določi okroglo obliko.

```
<g class="node node--leaf" name="zvok1" style="">
  <circle class="" transform="translate(38.21,-132.39)" r=
      "65.01" style=""></circle>
```

```
3    <image class="leafImg" x="-65.01" y="-65.01" transform="
        translate(38.21,-132.39)" clip-path="url(#i1)"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:href="img/234.jpg" width="130.03" height="
        130.03"></image>
4  </g>
```

Naslednji korak vizualizacije je transformacija. Tukaj krogom določimo radij in translacijo. Transformacija je v osnovi sestavni del predloge in se zgodi v funkciji *zoomTo(v)*. Podamo ji podatke o položaju ter velikosti korena, nato pa sama izračuna vse potrebne podatke za vsa vozlišča.

Stopnja podrobnosti

Množica posnetkov je sorazmerno velika, zato so posamezna vozlišča precej majhna, da se lahko vsa prikažejo na ekranu. Vsak list oz. posnetek vsebuje sliko povzročitelja zvoka, katere velikost je odvisna od velikosti lista. Ob tem nastane množica majhnih pik in vsebine ni mogoče razločiti. Nerazločna vsebina ni uporabniško prijazna, zato smo v začetnem prikazu prikrili podrobnosti, ki se nahajajo nižje od druge stopnje hierarhičnega drevesa.

Podrobnosti smo prikrili z naključno izbranimi slikami iz določenega poddrevesa. Slike so nekakšna maska, ki se s potovanjem v globino skrije oziroma se pokaže, ko se vrnemo na osnovni nivo. Obliko maske ponovno določajo objekti *clipPath*. Slike, ki jo sestavljajo, se ob vsaki osvežitvi strani določijo naključno, s klikanjem nanje pa lahko tudi pregledujemo, kakšni zvoki se nahajajo pod njo.

Predvajanje zvoka

Diplomska naloga govori o vizualizaciji zvoka. Zgolj grafična predstavitev podatkov ne zadovolji vseh potreb, zato implementiramo tudi predvajanje izbranega zvoka. Zvok se vedno nahaja v listu drevesa. Ko uporabnik klikne nanj, se naloži v predvajalnik in se začne predvajati. Za predvajanje in prikaz podrobnosti posnetkov smo napisali funkcijo *selectSound(sound)*.

Funkcije prejme objekt, ki vsebuje podatke o posnetku, jih prebere in pripne na ustrezne elemente spletne strani. Posodobijo se podatki v tabeli, slika zvoka in zvok v predvajalniku. Predvajanje se sproži tik pred koncem funkcije.

Gumb za korak nazaj

Pri gradnji aplikacije vedno želimo doseči, da bo karseda prijazna do uporabnika. Ena izmed dobrih praks je možnost razveljavitve zadnje uporabniške akcije. Za ta namen smo v zgornji levi kot dodali gumb, ki vizualizacijo postopoma pomika iz trenutnega pogleda proti osnovnemu.

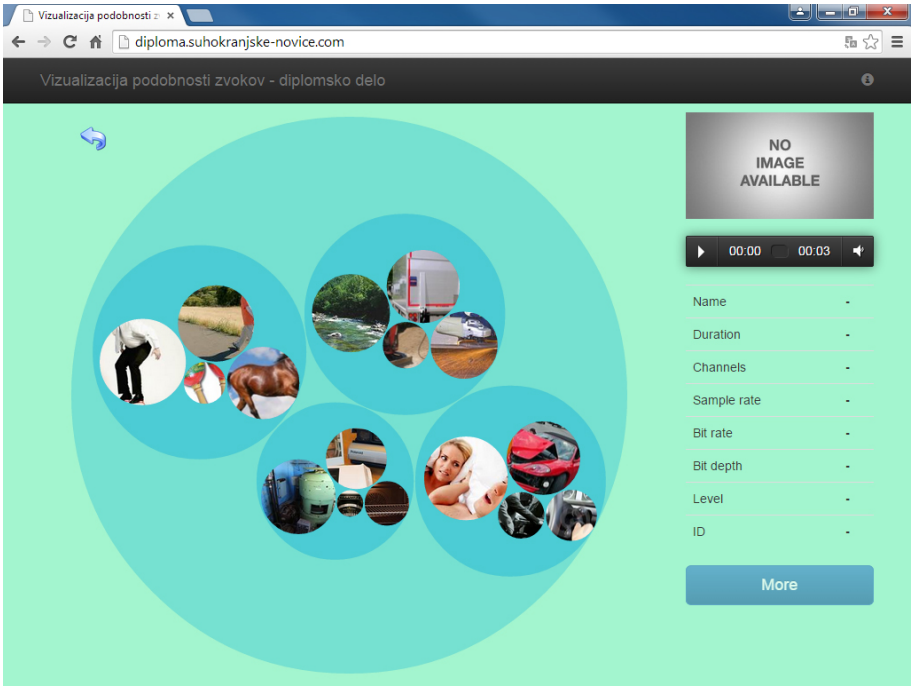
Gumb je fiksna slika, pripeta v vsebovalnik SVG, ki se odziva na klik. Ob kliku se pomakne za stopnjo višje v hierarhičnem drevesu, hkrati pa ustavi morebitno predvajanje posnetka ter poenostavi prikaz podatkov na desni strani vizualizacije.

4.6 Predstavitev in opis delovanja aplikacije

Spletna aplikacija, ki smo jo izdelali v praktičnem delu diplomske naloge, je prikazana na posnetku zaslona 4.1. Aplikacija prikazuje podobnosti med zvoki in podatke o izbranem zvoku s sliko povzročitelja ter omogoča poslušanje posnetka.

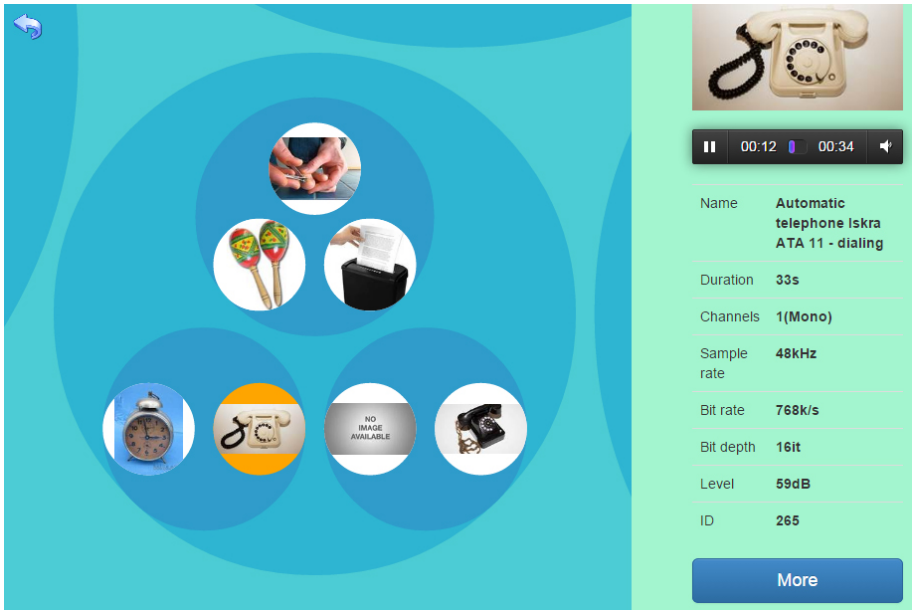
V začetnem pogledu prikazuje aplikacija štiri skupine. Skupine oz. krogi so prikriti s sliko, ki je naključno izbrana iz posamezne skupine in predstavlja enega izmed zvokov. Na desni strani je prostor za sliko pozneje izbranega zvoka, predvajalnik ter tabelo za tekstovne podatke. V zgornjem levem kotu je naslov, na desni strani pa gumb za odpiranje pojavnega okna z osnovni informacijami o aplikaciji. Aplikacija se odziva na naslednje uporabnikove akcije:

- prehod miške čez sliko poveča sliko;



Slika 4.1: Osnovni pogled na aplikacijo

- klik na sliko v začetnem prikazu menja sliko, ki zakriva podrobnosti določene skupine;
- klik na skupino približa oz. oddalji skupino. Akcijo napoveduje kazalec miške in je odvisna od trenutnega stanja vizualizacije.
- Klik na objekt s sliko v povečanem načinu le-tega osvetli, sproži predvajanje ter v tabeli prikaže vse podatke o izbranem posnetku. Če obstaja tudi zunanja povezava z več podrobnosti, se omogoči gumb „More“;
- predvajanje lahko kadarkoli prekinemo s klikom na gumb „stop“ ali gumb „nazaj“, ki se nahaja v zgornjem levem kotu vizualizacije.
- Če vizualizacija ni v osnovnem pogledu, gumb „nazaj“ zmanjša stopnjo povečave za ena, ustavi morebitno predvajanje zvoka, pobriše tekstovne podatke v tabeli in onemogoči gumb „More“.



Slika 4.2: Pogled na aplikacijo z izbranim posnetkom

Poglavje 5

Zaključek

V diplomskem delu smo pridobili podatke z zvočnih posnetkov in jih vizualizirali. Vizualizacija, ki podpira nekaj uporabniške interakcije, prikazuje podobnosti med obdelanimi zvoki. Za boljšo uporabniško izkušnjo smo posnetkom dodali tudi fotografije in omogočili predvajanje.

Najprej smo pripravili delovno množico 295 posnetkov, ki jo delno sestavljajo posnetki strojev iz evropskih tehniških muzejev. S pomočjo knjižnice ESSENTIA smo za vsak posnetek izračunali 27 nizkonivojskih značilk in tako pridobili 295 različnih vektorjev. Nato smo z orodjem Matlab izračunali kosinusne razdalje med vektorji. Rezultat je bila simetrična matrika podobnosti, nad katero smo v nadaljevanju z istim orodjem izvedli grozdenje. Hkrati smo po standardu JSON rezultate grozdenja izpisovali v datoteko. Tako pripravljeni podatki so bili primerni za izdelavo grafične vizualizacije, ki smo jo izvedli s pomočjo elementov SVG, generiranih s knjižnico D3.js. S to knjižnico smo izdelali tudi uporabniško interakcijo.

Vizualizacija je spletna aplikacija, ki ni odvisna od operacijskega sistema. Za delovanje potrebujemo zgolj spletni brskalnik z dostopom do interneta. Celotna koda se izvaja na uporabnikovi strani, torej v brskalniku. Izkazalo se je, da aplikacija najbolje deluje v brskalnikih Chrome in Safari, sledi Internet Explorer ter z nekaj težavami še brskalnik Firefox. Pri slednjem je slabše delovanje grafike znan problem, ki še vedno čaka na rešitev. Aplika-

cija zadovoljivo deluje tudi na tabličnem računalniku in sodobnem pametnem telefonu.

Aplikacija je prva tovrstna različica, zato je nekaj prostora tudi za izboljšave in nadaljnji razvoj. Ena izmed možnosti je dodelava uporabniške interakcije na zaslonih na dotik. Druga možnost je izboljšava stopnje podrobnosti, ki bi optimizirala prikaz pri večjih zvočnih zbirkah. Zanimivo bi bilo uporabniku ponuditi interaktivno izbiranje značilk, ki vplivajo na podobnosti, ali celo dodajanje svojih zvočnih posnetkov. Možnosti je veliko, zato bo nadaljnji razvoj še kako zanimiv.

Literatura

[1] Bootstrap framework. http://getbootstrap.com/ (31. 1. 2015).

[2] Brackets. http://brackets.io/ (31. 1. 2015).

[3] Css. http://www.w3schools.com/css/css_syntax.asp (31. 1. 2015).

[4] Data-driven documents. http://d3js.org/. (31. 1. 2015)

[5] Essentia. http://essentia.upf.edu/ (31. 1. 2015).

[6] Formater. http://www.formater.de/ (31. 1. 2015).

[7] Hierarchical clustering. http://www.mathworks.com/help/stats/hierarchical-clustering.html (31. 1. 2015).

[8] Live plasma. http://www.liveplasma.com/ (31. 1. 2015).

[9] Matlab. http://www.mathworks.com/products/matlab/ (31. 1. 2015).

[10] Music map. http://www.music-map.com (31. 1. 2015).

[11] Musicovery. http://musicovery.com/ (31. 1. 2015).

[12] Neptune. http://www.cp.jku.at/projects/nepTune/ (31. 1. 2015).

[13] Play som. http://www.ifs.tuwien.ac.at/mir/playsom.html (31. 1. 2015).

[14] Sonic visualiser. http://www.sonicvisualiser.org/ (31. 1. 2015).

[15] Svg. http://www.w3schools.com/svg/svg_intro.asp (31. 1. 2015).

[16] Work with sounds. http://www.workwithsounds.eu/ (31. 1. 2015).

[17] Michael Friendly and Daniel J Denis. Milestones in the history of thematic cartography, statistical graphics, and data visualization. *U RL http://www. datavis. ca/milestones*, 2001.

[18] Barbara Kermavner. *Gručenje učencev pri pouku računalništva*. Diplomsko delo, B. Kermavner, 2011.

[19] Miha Krašovec. *Prepoznavanje sestavov v posnetkih ljudske glasbe*. Diplomsko delo, Univerza v Ljubljani, 2011.

[20] Anita Shen Lillie. *MusicBox: Navigating the space of your music*. PhD thesis, Massachusetts Institute of Technology, 2008.

[21] Anže Rozman. *Novosti HTML5 in grafični vmestnik canvas*. Diplomsko delo, Univerza v Ljubljani, 2010.

[22] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[23] Marija Štokelj. *Uporaba SVG pri razvoju odzivnih spletnih strani*. PhD thesis, Univerza v Ljubljani, 2014.

[24] Tadej Vrtovec. *Vizualizacija zvočnih zbirk*. Diplomsko delo, Univerza v Ljubljani, 2014.

[25] Claus Weihs, Klaus Friedrichs, Markus Eichhoff, and Igor Vatolkin. Software in music information retrieval. In *Challenges at the Interface of Data Analysis, Computer Science, and Optimization*, pages 421–432. Springer, 2012.

[26] Frans Wiering. Can humans benefit from music information retrieval? In *Adaptive Multimedia Retrieval: User, Context, and Feedback*, pages 82–94. Springer, 2007.